Building data.eurecom.fr

Semester Project Report

Anne-Elisabeth Gazet

Fall 2011

Supervisors:
Raphaël Troncy
Ghislain Atemezing

EURECOM
Multimedia Department

Contents

Introduction

The web as we know it today mainly consists of documents that are linked together. We are now moving to a new era, where raw data is being published on the web as *Linked Data*, and woven into the *Web of Data* or *Semantic Web*.

> "The vision of the Semantic Web is to extend principles of the Web from documents to data. Data should be accessed using the general Web architecture using, e.g., URI-s; data should be related to one another just as documents (or portions of documents) are already. This also means creation of a common framework that allows data to be shared and reused across application, enterprise, and community boundaries, to be processed automatically by tools as well as manually, including revealing possible new relationships among pieces of data."
>
> from the W3C Semantic Web FAQ[1]

In order to achieve these goals, new technologies were developed:

- RDF, for Resource Description Framework, is the main building block of the Semantic Web. It's a simple interchange format, where data is represented as triples of the form (subject, predicate, object) ;
- OWL (Web Ontology Language) and RDFS (RDF Schema) are vocabularies for describing RDF properties and classes ;
- SPARQL (SPARQL Protocol and RDF Query Language) is a protocol and query language for semantic web data sources ;
- SPARQL Update, a companion language for SPARQL, enables modifying a data source.

More information on the Linked Data paradigm can be found in [1].

[1]http://www.w3.org/2001/sw/SW-FAQ

Motivation for the project

There is a trend to increase government transparency by releasing more and more public sector information on the web as linked data. The first initiative of the kind was data.gov.uk in the United Kingdom, which was followed by similar projects around the world.

Universities and schools also generate a lot of data about students, promotions, professors, courses, publications, departments, rooms, schedules, exams, etc. The goal of this project is to take all this data generated by EURECOM, transform it in semantic formats (RDF), interlink it with other data (from other universities and datasets) and publish the whole as linked data in order to develop showcase applications that provide useful services to students and professors.

Chapter 1

Similar projects

This project was inspired by recent similar initiatives in other universities. We studied them in order to see what kind of data they made available, which approach they took in order to publish their data, and what kind of applications were built thanks to the newly exposed data.

These are the projects we studied in particular:

- the Open University's LUCERO (Linking University Content for Education and Research Online) project was the first initiative to expose public information from a university as Linked Open Data. As stated on the project's blog[1], LUCERO aim in particular at answering the following questions: *"What are the workflows, business processes, policies and technologies needed to expose the Open University and related digital content as linked data?", "How can we integrate linked data technology in a sustainable way to support the research and educational activities of a Further or Higher Education organisation?"* Their portal is located at `http://data.open.ac.uk/`.

- the University of Southampton also offers an open data service. The goal of this project is to take all data that the university has and that is not confidential, and make the data *"available in a structured way with a license which allows reuse* [so that] *our members, or anyone else, can build tools on top of it without needless bureaucracy"*. The portal is located at `http://data.southampton.ac.uk/`.

- the University of Muenster also has a linked open data project, in early beta stage. It is reachable at `http://data.uni-muenster.de/`.

- the University of Oxford's similar project is also in a very early stage. It is located at `http://data.ox.ac.uk/`.

[1]http://lucero-project.info/lb/

1.1 Exposed data

The domains covered by the published datasets are usually:

- research outputs
- course descriptions
- course material
- audio and video material
- internal organization
- geographical data about buildings
- projects

The University of Southampton also published more “exotic” data about services, student statistics, published accounts, etc...

1.2 Vocabularies used

Among the vocabularies used to describe the data, we can find those that are standards or de facto standards: Dublin Core[2] terms for generic metadata, Friend Of A Friend[3] to describe persons, Good Relations[4] for e-commerce, the Basic Geo Vocabulary[5] for geographical resources.

Alongside those, we found some less common vocabularies, often more specific to the domain covered by university data. Among those,

- AIISO[6] and Courseware[7] ontologies are used to describe courses.
- AIISO is also used, like the Organization ontology[8], to describe the universities’ internal organization.
- LODE[9] and the Event Ontology[10] are used to describe events, such as conferences.
- the Data Cube ontology[11] is used to give numerical data.

[2]http://dublincore.org/documents/dcmi-terms/
[3]http://xmlns.com/foaf/spec/
[4]http://www.heppnetz.de/projects/goodrelations/
[5]http://www.w3.org/2003/01/geo/
[6]http://vocab.org/aiiso/schema
[7]http://courseware.rkbexplorer.com/ontologies/courseware
[8]http://www.epimorphics.com/public/vocabulary/org.html
[9]http://linkedevents.org/ontology/
[10]http://motools.sourceforge.net/event/event.html
[11]http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/vocab/cube.ttl

- BIBO[12] or the ISBD vocabulary[13] are used to describe bibliographic resources.

1.3 Applications

Common applications are maps of the buildings, phone books, help for finding and choosing courses.

At Southampton University, a map of the campus[14] is being developed that shows very diverse information about services, catering, bus stops, culture and entertainment...

At the Open University, an Android application called wayOU, for "where are you in the OU?" allows users to share their locations on the campus. A contest was also organized to encourage people to develop new applications using their data.

[12]http://bibliontology.com/specification
[13]http://iflastandards.info/ns/isbd/elements/
[14]http://opendatamap.ecs.soton.ac.uk/

Chapter 2

Data model and design of our ontology

Publishing data in RDF format does not make it automatically useful and reusable. It is very important to choose carefully the way the data will be modeled, and to document this. This is the most time-consuming task in such a project, but it is well worth it: a well-designed data model will help future application developers consume the data, and combine it with external datasets.

2.1 Available data

One of the difficulties with this project is that we had to think about the data model *before* actually getting access to the data, and therefore, before knowing exactly what data would be exposed. To have an idea of what data was stored at EURECOM, we had a look at the internet and intranet sites of the institute. The information available there is mainly about:

- people (teachers, researchers, doctoral students, staff members),
- research outputs,
- courses.

Assuming these would be the datasets we would access later on, we designed an RDF data model.

2.2 Choice of external vocabularies

To achieve re-usability, it is paramount to represent data using vocabularies that are used by other projects as well. This way, the data will be easier to consume for existing applications. Furthermore, the fact that a vocabulary was accepted by the community as a de facto standard is a good hint that this vocabulary is well designed, and well fitted to its domain.

Yet, since there are still relatively few universities publishing their data as Linked Data, we could not rely only on popularity to make choices. Besides, each dataset being unique, there are relationships that we wanted to represent, to which we did not find any equivalent in the other projects' datasets. In this case, we used tools such as Schema-Cache[1], Schemapedia[2] and Sindice[3] to see if there existed any term we could use.

Alongside popularity, the other criteria we took into account to select vocabularies are:

- the similitude between the concepts and terms present in the vocabulary, and the concepts needed to represent our data ;
- whether the vocabulary was created as part of a bigger project: this means it is the result of a consensus rather than the vision of a sole individual on the domain.

In the end, we selected the following vocabularies:

- FOAF[4]: the Friend Of A Friend vocabulary defines terms for describing persons and their relations to other people and objects. We use it to describe people at EURECOM.
- Dublin Core terms[5]: this vocabulary is a set of generic metadata terms whose purpose is to describe numeric and physical resources. We use for example the predicate `dc:creator` to represent the relationship between a document and its creator.
- Participation[6]: The participation ontology is a simple model for describing the roles that people play within groups. We subclass the class `part:Role` in the REVE ontology, to describe roles played by people inside EURECOM.
- AIISO[7]: The Academic Institution Internal Structure Ontology provides classes and properties to describe the internal organizational structure of an academic institution. We use the `aiiso:Course` and `aiiso:KnowledgeGrouping` classes to define courses and tracks.

[1]http://schemacache.com/
[2]http://schemapedia.com/
[3]http://sindice.com/
[4]http://xmlns.com/foaf/spec/
[5]http://dublincore.org/documents/dcmi-terms/
[6]http://vocab.org/participation/schema
[7]http://vocab.org/aiiso/schema

- BIBO[8]: The Bibliographic Ontology describe bibliographic things on the semantic Web in RDF. It has been inspired by many existing document description metadata formats. We use it to describe the articles in EURECOM's scientific publications repository.
- LODE[9]: The ontology for Linking Open Descriptions of Events is an ontology for publishing descriptions of historical events as Linked Data, and is designed to be compatible with other event-related vocabularies and ontologies. We use the predicates `lode:atTime`, `lode:atPlace` and `lode:involvedAgent` to describe course sessions.
- OWL-Time[10]: This ontology provides a vocabulary for expressing facts about topological relations among instants and intervals, together with information about durations, and about datetime information. We use it to describe the temporal aspects of course sessions.
- Rooms[11]: It's simple vocabulary for describing the rooms in a building, which we use to describe the rooms and buildings where courses take place.

Note that `dc`, `part`, `aiiso` and `lode` are the corresponding prefixes for the vocabularies' associated namespaces. The same notation is used in the rest of this report.

2.3 REVE, the Research and Education Vocabulary for EURECOM

Some of the concepts we need to represent the data are specific to EURECOM, so we defined a set of new terms. When it was possible, we defined these terms as extensions of existing terms.

The namespace for the vocabulary is `data.eurecom.fr/ontology/reve`. In the rest of this report, a name of the form `:termName` designates an RDF term in the REVE namespace.

The vocabulary is specified using the RDFS and OWL languages. The latest specification can be found in RDF format at the following address: `https://github.com/aegazet/REVE-Ontology`

You will find a graph representing the ontology as annex A of this report.

2.3.1 Example modeling issue: what is a course ?

One of the main modeling issues we encountered was due to a particularity of the french language: the word "cours" can mean many different things, even

[8]http://bibliontology.com/specification
[9]http://linkedevents.org/ontology/
[10]http://www.w3.org/TR/owl-time/
[11]http://vocab.deri.ie/rooms

when we restrict ourselves to the context of a school. For instance, it can designate a lecture, or a course. We also had to make the distinction between a course which took place in a given semester, and the general subject of the course. E.g., "WebSem - Spring 2010" and "WebSem - Spring 2011" are two modules which both treated the same subject — an introduction to the Semantic Web — but they took place in different times, and different people were registered for them.

In such cases, it is essential to discuss the issue with experts on the domain we try to model. Here we asked Alexia Cepero, the Student's Pedagogy Officer of EURECOM, what was the correct representation. This led us to define the two following classes:

- Course: a teaching unit. A course is composed of several course sessions.
- Course session: a course session is a punctual event on which teacher and students gather, for a given course.

Counter intuitive as it might seem, there is no concept representing the unity of subject between, e.g., "WebSem - Spring 2010" and "WebSem - Spring 2011". But in the data we publish, we can still hint at a relationship between the two instances, for example by stating they have the same `aiiso:code` attribute.

2.3.2 Representing an n-ary relation

Another issue we encountered, but was easier to solve, resulted from an assumption we made on the data which turned out to be false. We had assumed the credits one can earn by successfully completing a course only depended on the course itself. It turns out that it also depends on the track of study the student is following at EURECOM.

In other words, what we had modeled as a binary (course, credit) relationship is actually a ternary (course, credit, track) relationship. The minor trouble with this, is that such a relationship cannot be represented with a single triple.

This is a common situation in ontology modeling. Good modeling practices have therefore been identified by the community: ontology patterns for representing n-ary relations in RDF and OWL are presented in a W3C working group note[12]. We picked the first pattern described in this note, which consists in creating a new class and n new properties to represent the n-ary relation. Figure 2.1 represents the resulting class and its properties.

[12] http://www.w3.org/TR/swbp-n-aryRelations/

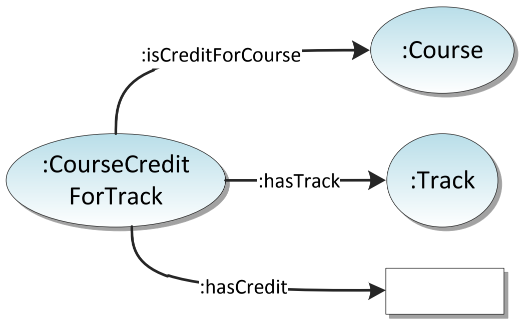


Figure 2.1: The class `:CourseCreditForTrack` and its properties

2.4 First data model

This first model was developed having in mind the data we had observed on the various sites of the school. Therefore, it is much more complete than what has been implemented so far, mainly because a lot of data has not been exposed yet. This preliminary model is documented here in the hope that it will be useful for future developments of the project.

You will find as annexes several graphs illustrating this model. Each graph is focused on a particular domain: persons, documents.

Persons

This model was inspired by data that is visible on EURECOM's public staff directory.

Documents

This model is mainly a subset of the Bibliographic Ontology. The only question raised by this part of the model concerns the ordered list of authors of an article.

Representing a list of authors

It seems natural to use BIBO's `bibo:authorList` predicate to represent an article's list of authors. But its usage is not straightforward: the range of this predicate is the union of the classes `rdf:Seq` and `rdf:List`. Both of these classes are *container* classes, that is to say they were designed to represent sets of objects. This is not an easy task with triples. It is generally advised not to use container classes in RDF at all, unless the order of the elements is important. Here the order does matter, so we had a closer look at those classes to make our choice:

- `rdf:List` is a linked list: instances of this class have an `rdf:first` attribute pointing at the *head* of the list, and an `rdf:rest` attribute pointing at the *tail* of the list (another `rdf:List`). The end of the list is indicated by setting the `rdf:rest` attribute to `rdf:nil`.

- `rdf:Seq` is more like a table: an instance of this class has attributes of the form `rdf:_1`, `rdf:_2`, ... `rdf:_n` pointing at the elements of the list.

The two approaches have upsides and downsides. Using `rdf:List`, you can indicate exactly what the elements of the list are, which you cannot with `rdf:Seq`. Indeed the Semantic Web is built on an *open world assumption*: the absence of a statement does not mean that the statement is false. In other words, you can state that an `rdf:Seq` has *at least* n elements, but you cannot state that there are no other elements.

On the other hand, if you use `rdf:List` to represent your data, you will need a lot of instances of `rdf:List`. This makes the structure of the data complicated, and in general hard to query. The more so as, since it makes no sense to give a globally unique identifier to every `rdf:List` in your data, this approach leads to creating *blank nodes* for them. Blank nodes are nodes without a URI, and it is good practice to avoid them whenever possible, as stated in [1]:

> "The scope of blank nodes is limited to the document in which they appear, meaning it is not possible to create RDF links to them from external documents, reducing the potential for interlinking between different Linked Data sources. In addition, it becomes much more difficult to merge data from different sources when blank nodes are used, as there is no URI to serve as a common key."

This motivated our final decision to use `rdf:Seq` to represent the list of authors.

2.5 Present state of the data

As stated in 2.4, the data that has been translated in RDF so far in this project only fits a subset of the general data model we imagined. The graph in annex D represents this subset. As you will see, the bibliographic part is much more simple than the previous model. This is mainly due to the poor quality of metadata available for now.

To start consume data now, application developers can refer to this graph to know what triples they can find in the triplestore. This kind of documentation is essential to make data reusable. Sadly, as we looked around for similar initiatives, we noticed it is often missing, even in larger scale projects. Let us remind here that publishing data in RDF format does not make it automatically reusable: if there is no information regarding the data model and the vocabulary used, the application developer has no way of guessing what information is available, and how to retrieve it.

Chapter 3

Data conversion

Once the data model has been defined, we can move on to a more concrete stage: the translation of data into RDF.

3.1 Getting the data

In the early stages of the project, the plan was to give us access to a dump of a relational database containing the interesting information. Yet for various reasons (practical, legal, ...) it was decided after a few weeks that the IT department would instead set up a web service serving JSON files for us to process.

In the weeks preceding this change of strategy, we had started to get familiar with R2RML, a language for expressing mappings from relational databases to RDF datasets[1]. We tested in particular db2triples[2], an open-source implementation of R2RML by the french firm Antidot. In the end, this was not used for the project, but could be very useful for future developments.

The JSON files served by the web service have the following format:

```
{"status":"successful",
"request_uri":"http:\/\/intranet-v2.eurecom.fr\/data\/...",
"label":"Courses list - 2004",
"count":0,
"response":[]}
```

The `status` attribute states whether or not the file could be retrieved successfully. The data is in the `response` attribute. The format of the response depends on the file. The different types of files presently available are:

[1]http://www.w3.org/TR/r2rml/

[2]http://blog.antidot.net/2011/10/07/db2triples-une-implementation-de-r2rml-et-directmapping-en-open-source/

- course: files describing courses contain the full and abbreviated names of the courses, the list of teachers for a given course, its abstract, ...
- course session: files describing course sessions repeat some of the information given in course files, and contain info on the sessions (date and hour, room and teacher involved)
- publications: these files contain articles' title, list of authors, the name of the conference where an article was presented, as well as the date and place of the conference
- teacher, researcher, doctor, phd: these files describe people in terms of the *role(s)* they hold or have held at EURECOM

3.2 Building RDF triples with rdflib

3.2.1 Why Python?

As Python is a very popular scripting language, there exist good tools to work with RDF in Python. Besides, with Python it is easy and fast to test things, and easy to work with JSON files. The drawback of using a scripting language rather than, say, Java, or any other compiled language, is a lesser speed of execution. But at this stage of the project, the conversion is done once and for all – it is more about testing feasibility – so we did not worry much about execution speed, and chose to use Python and the rdflib library to make the data conversion.

3.2.2 The rdflib library

The rdflib package is intended to provide core RDF types and interfaces for working with RDF.

The primary interface that rdflib exposes for working with RDF is `Graph`. rdflib graphs have ordinary set operations (e.g. `add()` to add a triple) plus methods that search triples and return them in arbitrary order. They are best thought of as a set of 3-item triples.

Elements of a triple are created using the following classes:

- `URIRef` for URI references
- `Literal` for literals (datatypes and language tags are supported)
- `BNode` for blank nodes

With extensions provided in the rdfextras library, it is possible to store the triples directly in a triplestore. Here, we used the serialization capacities of rdflib: a graph is easily serialized in RDF/XML format, and saved in a file.

Moreover, rdflib provides a `parse()` method which not only loads a graph from a file, but can also retrieve files from the web.

rdflib also has a convenient `Namespace` class, with which fully qualified URIs in the namespace can be constructed by accessing a `Namespace` object like a dictionary:

```
>>> DCT = rdflib.Namespace("http://purl.org/dc/terms/")
>>> DCT["creator"]
rdflib.term.URIRef('http://purl.org/dc/terms/creator')
```

More information on how to use rdflib is available on the library's documentation pages[3].

3.2.3 URI scheme

We took the following conventions to assign URIs to the entities described in the data:

- the base URL for the dataset is `http://data.eurecom.fr/`
- every resource gets a URI of the form:
  `http://data.eurecom.fr/<domain>/<id>`
  where `<domain>` corresponds to the type of the entity, and is one of the following:
  - `course`
  - `semester`
  - `people`
  - `publication`
  - `session`
  - `room`
  - `track`
  - `role`
  - `department`
  - `event`
- for the `<id>` part of the URI, we generally used the integers that identify the entities in the original database records, and are present in the JSON files.
- for the conferences, there are no such identifiers in the database, only strings of characters describing the event (e.g. "ISM 2011, IEEE International Symposium on Multimedia, December 5-7, 2011, Dana Point, CA, US"), so the identifier is the MD5 hash of this string.

[3]http://readthedocs.org/docs/rdflib/en/latest/

3.2.4 Conversion strategy

The conversion process is pretty simple. It boils down to reading sequentially the input JSON files, and apply the "rules" defined in our RDF data model, and in the URI scheme.

First thing to do is to get a file from the web service, using the `urllib` module. The file is then parsed into a Python dictionary using the `json` module, and the elements of the `response` array are parsed individually.

In general, one RDF triple is generated for every attribute of every element, then added to one of the following graphs: courses, sessions, semesters, tracks, people, roles, publications, events, rooms, departments. This splitting of the triples in several graphs results in the creation of several RDF files, each containing information on a particular domain.

3.3 Linking data to an external source: GeoNames

At this point of the data conversion, we have information about articles that were presented at conferences all around the world. But we do not have much geographical information about the conferences. The JSON files contain, for each article, the name of the city where the conference took place, as well as the name and the ISO two-letter code for the country:

```
city": "Cape Town",
"country": {
    "id": "country/za",
    "shortlabel_en": "SOUTH AFRICA",
    "longlabel_en": null
}
```

This information is enough to do some geocoding[4], so as, for example, to be able to represent the conferences' locations on a map. The geocoding service we used is GeoNames[5], a popular geographical database available and accessible through various Web services. Each geographical entity it describes has a unique integer assigned to it, as well as a URI to be used in semantic web applications. GeoNames data are linked to DBPedia, the project publishing content from Wikipedia as linked data.

The geocoding service is available through a REST API. This is the query to retrieve information about Cape Town, South Africa:

```
http://ws.geonames.org/searchJSON?
q=%22Cape%20Town%22&country=za
```

[4]from Wikipedia: *Geocoding is the process of finding associated geographic coordinates (often expressed as latitude and longitude) from other geographic data, such as street addresses, or zip codes (postal codes).*

[5]http://www.geonames.org/

And this is the response, in JSON format:

```
{
  "totalResultsCount": 43,
  "geonames": [
    {
      "countryName": "South Africa",
      "countryCode": "ZA",
      "lng": 18.423218,
      "name": "Cape Town",
      "geonameId": 3369157,
      "lat": -33.925839,
      ...
    }
  ]
}
```

From this response, we extract the geonameID, use it to construct the URI for the place, using the scheme: `sws.geonames.org/<geonameID>`, and use this URI reference in new triples. We also store the latitude and longitude in RDF format for later use. Indeed, GeoNames does not provide an official SPARQL endpoint, so every application developer willing to get their data directly in RDF format has to replicate the data locally (or to use an unofficial SPARQL endpoint.)

3.4 Loading triples in a triplestore using SPARQL Update

The last step of the conversion consists in automating the loading of the triples into a triplestore.

To do so, we used the SPARQL Update language. It is not a W3C recommendation yet, but should soon become one. It has been mature for some time now. Consequently, most of the popular triplestores support it.

The principle is simple: the update request is sent to the SPARQL endpoint via the HTTP POST method by URL encoding the parameters. Here we use the LOAD command to load a local file into the triplestore:

```
LOAD <file:// ... > INTO GRAPH <http://data.eurecom.fr/rooms/>
```

The triplestore is now loaded with the newly generated triples, and ready to answer queries.

Chapter 4

Showcase application : a map of EURECOM publications

There are no killer app for the Semantic Web. It rather enables to build many new services, and then it is up to the programmer to get inspiration from the data.

To demonstrate what can be done with the newly generated triples, we developed a map of the world with pins showing every conference where EURECOM researchers presented a paper. This was made easy thanks to the SIMILE Exhibit project. Exhibit[1] is a tool to create data visualization web pages. With a bit of experience, one can create an Exhibit page within an hour or two.

The capital thing that is required to use Exhibit, is to prepare your data: it has to be in a particular JSON format. The minimal Exhibit JSON file contains an array named `items`, whose elements have at least a `label` and a `type` attributes:

```
{  items: [
     {  label:  "Jane Smith",
        type:   "Person"  }
   ]
}
```

Some data importing services are proposed, through the SIMILE Babel project[2]. Unfortunately, the conversion from RDF/XML (or another serialization like N3) to Exhibit JSON is poorly made. In particular, it fails to use the URI of an RDF term as the identifier for that resource in the JSON file. As a consequence, it breaks all the *links* in the linked data. So we resorted to implementing the conversion ourselves, in Python again, with help from the `SPARQLWrapper` module.

[1]http://simile-widgets.org/exhibit/
[2]http://simile.mit.edu/babel/

First, all data about papers that were presented at conferences are retrieved by issuing the following SPARQL query:

```
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX dct:<http://purl.org/dc/terms/>
PREFIX foaf:<http://xmlns.com/foaf/0.1/>
PREFIX lode:<http://linkedevents.org/ontology/>
PREFIX geo:<http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX bibo:<http://purl.org/ontology/bibo/>

SELECT DISTINCT
  ?pub ?title ?date ?conf
  ?conf_title ?lat ?long ?author
WHERE {
  ?pub rdf:type foaf:Document .
  ?pub dct:title ?title .
  ?pub dct:date ?date .
  ?pub bibo:presentedAt ?conf .
  ?conf rdfs:label ?conf_title .
  ?conf lode:atPlace ?place .
  ?place geo:lat ?lat .
  ?place geo:long ?long .
  ?pub dct:creator ?author
}
```

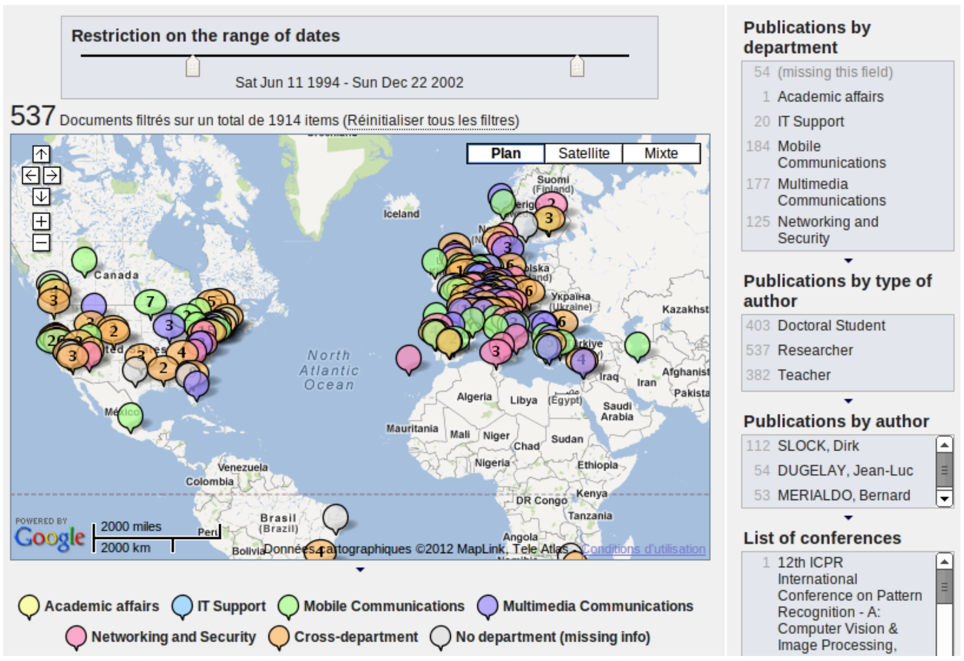
The response, in JSON format, is then transformed and put into a Python dictionary. A second query is issued to get information about the authors:

```
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX dct:<http://purl.org/dc/terms/>
PREFIX foaf:<http://xmlns.com/foaf/0.1/>
PREFIX part:<http://purl.org/vocab/participation/schema#>

SELECT DISTINCT
  ?author ?fn ?ln ?roletype ?depname
WHERE {
  ?pub rdf:type foaf:Document .
  ?pub dct:creator ?author .
  ?author foaf:firstName ?fn .
  ?author foaf:lastName ?ln .
  ?author part:holder_of ?role .
  ?role rdf:type ?roletype .
  OPTIONAL {
    ?role part:role_at ?dep .
    ?dep rdfs:label ?depname
  }
}
```

The response is parsed in the dictionary too. The dictionary gets enriched by some information specific for Exhibit to work with the data, then is serialized into a JSON file.

The resulting application can currently be viewed at the following address: `perso.telecom-paristech.fr/~gazet/map.html`



Restriction on the range of dates

Sat Jun 11 1994 - Sun Dec 22 2002

537 Documents filtrés sur un total de 1914 items (Réinitialiser tous les filtres)

Publications by department

54	(missing this field)
1	Academic affairs
20	IT Support
184	Mobile Communications
177	Multimedia Communications
125	Networking and Security

Publications by type of author

403	Doctoral Student
537	Researcher
382	Teacher

Publications by author

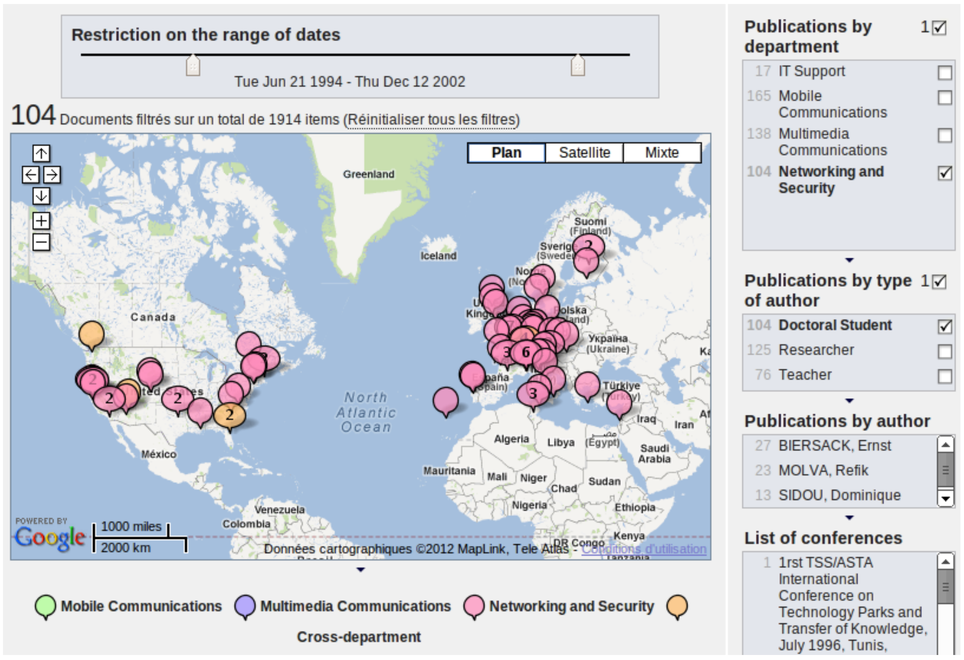
112	SLOCK, Dirk
54	DUGELAY, Jean-Luc
53	MERIALDO, Bernard

List of conferences

1	12th ICPR International Conference on Pattern Recognition - A: Computer Vision & Image Processing,

○ Academic affairs ○ IT Support ○ Mobile Communications ○ Multimedia Communications ○ Networking and Security ○ Cross-department ○ No department (missing info)

Figure 4.1: Screenshot of the application.
The lists on the right are *facets*: clicking on an element will apply a filter on the data displayed. The time range can also be adjusted with the slider on top of the map.



Restriction on the range of dates

Tue Jun 21 1994 - Thu Dec 12 2002

104 Documents filtrés sur un total de 1914 items (Réinitialiser tous les filtres)

Mobile Communications | Multimedia Communications | Networking and Security | Cross-department

Publications by department 1 ✓

17	IT Support	☐
165	Mobile Communications	☐
138	Multimedia Communications	☐
104	**Networking and Security**	✓

Publications by type of author 1 ✓

104	**Doctoral Student**	✓
125	Researcher	☐
76	Teacher	☐

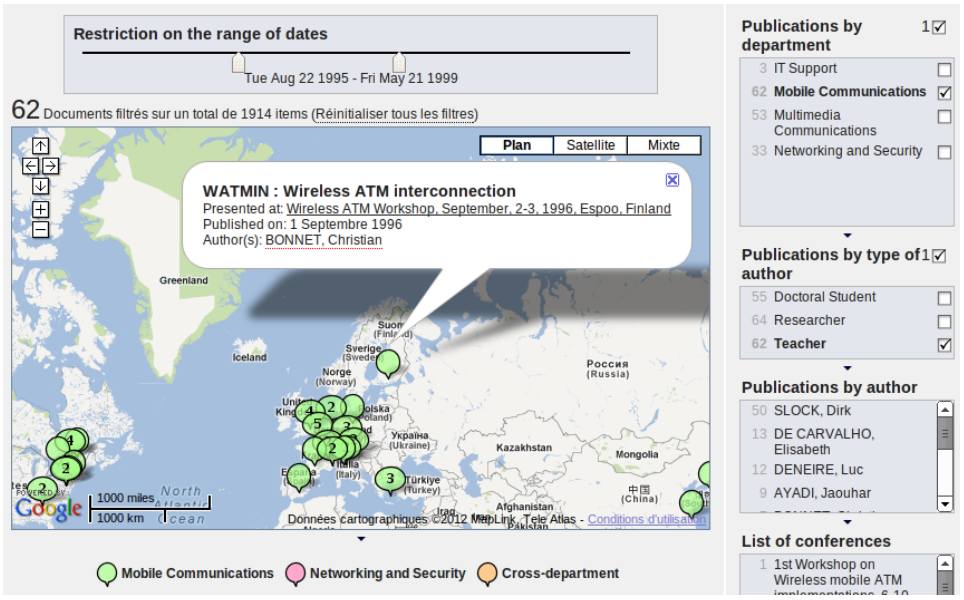
Publications by author

27	BIERSACK, Ernst
23	MOLVA, Refik
13	SIDOU, Dominique

List of conferences

1 1rst TSS/ASTA International Conference on Technology Parks and Transfer of Knowledge, July 1996, Tunis,

Figure 4.2: Screenshot of the application with filters on



Restriction on the range of dates

Tue Aug 22 1995 - Fri May 21 1999

62 Documents filtrés sur un total de 1914 items (Réinitialiser tous les filtres)

WATMIN : Wireless ATM interconnection
Presented at: Wireless ATM Workshop, September, 2-3, 1996, Espoo, Finland
Published on: 1 Septembre 1996
Author(s): BONNET, Christian

Mobile Communications | Networking and Security | Cross-department

Publications by department 1 ✓

3	IT Support	☐
62	**Mobile Communications**	✓
53	Multimedia Communications	☐
33	Networking and Security	☐

Publications by type of author 1 ✓

55	Doctoral Student	☐
64	Researcher	☐
62	**Teacher**	✓

Publications by author

50	SLOCK, Dirk
13	DE CARVALHO, Elisabeth
12	DENEIRE, Luc
9	AYADI, Jaouhar

List of conferences

1 1st Workshop on Wireless mobile ATM

Figure 4.3: Clicking on a pin displays information on the publication

Chapter 5

Future work

We think the work done for this project proves the feasibility and relevance of applying the linked data paradigm to university data at EURECOM. But there is still a lot to do to take advantage of the full potential of the Semantic Web at the school.

Improve on architecture

In the current state of the project, the RDF data is generated once and for all, then loaded into the triplestore. It does not evolve when the data delivered by the web service changes. There are different ways to improve this:

- a quick and dirty solution would be to crawl the web service regularly, generate all the triples over again, then update the triplestore with the result.
- a better approach would be to build an RSS feed on top of the web service, describing additions and changes made to the underlying database. An analysis of this feed would then enable updating the triplestore. This is the solution adopted by DBPedia[1].
- a further improvement would be to implement the PubSubHubbub protocol[2]. This protocol improves on the RSS feed mechanism, where the client usually polls the server at regular intervals to check for updates. In PubSubHubbub, clients subscribe to a feed on a *hub*. The feed publisher informs the hub of any updates, and the hub then sends the update to the subscribers. This mechanism allows for almost real-time updates, and a more efficient use of bandwidth.

[1]http://live.dbpedia.org/
[2]http://code.google.com/p/pubsubhubbub/

Expose more data

As more data will be exposed, more interesting applications will become possible. It would be nice for example, to get information on the projects the researchers are currently working on.

In the long run, the ideal would be of course to re-think the production of data entirely, so as to generate RDF data directly at the source, instead of going through a conversion process.

Clean up data

One of the benefits of exposing data, is that you get more people examining it, and discovering things that should not be. As we were working on the conversion, and then on the visualization, we noticed small strange things that were due to errors in the data. As more data is exposed, expect more people to notice and report such details that would otherwise go unnoticed. It is a good opportunity to clean up the data source.

There is also work to be done on the generated RDF: in the future, it will be interesting to determine whether flaws in the generated triples are due to flaws in the source of data, or flaws in the conversion. Then it has to be decided whether to fix the data at its source, or to take advantage of the semantic web technologies to do that.

An area that really needs cleaning up is the set of conferences. They are currently identified by a string looking like "ISM 2011, IEEE International Symposium on Multimedia, December 5-7, 2011, Dana Point, CA, US". The problem with this approach is that these strings were given manually by individual researchers. As a consequence, one conference often has several names, and articles that were presented at a same event are not recognized as such. It is paramount to reconcile identifiers that are meant to represent the exact same entity.

Alignment with other datasets

Linking datasets together is what makes the power of the Semantic Web. Creating links from our data to some well-connected datasets from the Linked Open Data cloud will make our data more visible, and more useful.

Once identifiers within our own domain will be reconciled, it will be great time to start interlinking our data with external datasets. This is often done thanks to the `owl:sameAs` predicate, which indicates that two identifiers refer to the same entity. It should be particularly interesting to see how to match our identifiers for people and documents with those from big bibliographic databases like DBLP.

Build more applications

Time did not permit to implement all the ideas we had to use the data:

- a track picker, to help students choose their track at EURECOM according to the courses they want to take
- a map of the buildings, with functionality to locate people or see which rooms are free for a meeting at some time
- a social graph of researchers, based on their common publications and projects (inspired by a similar application from the Open University[3])

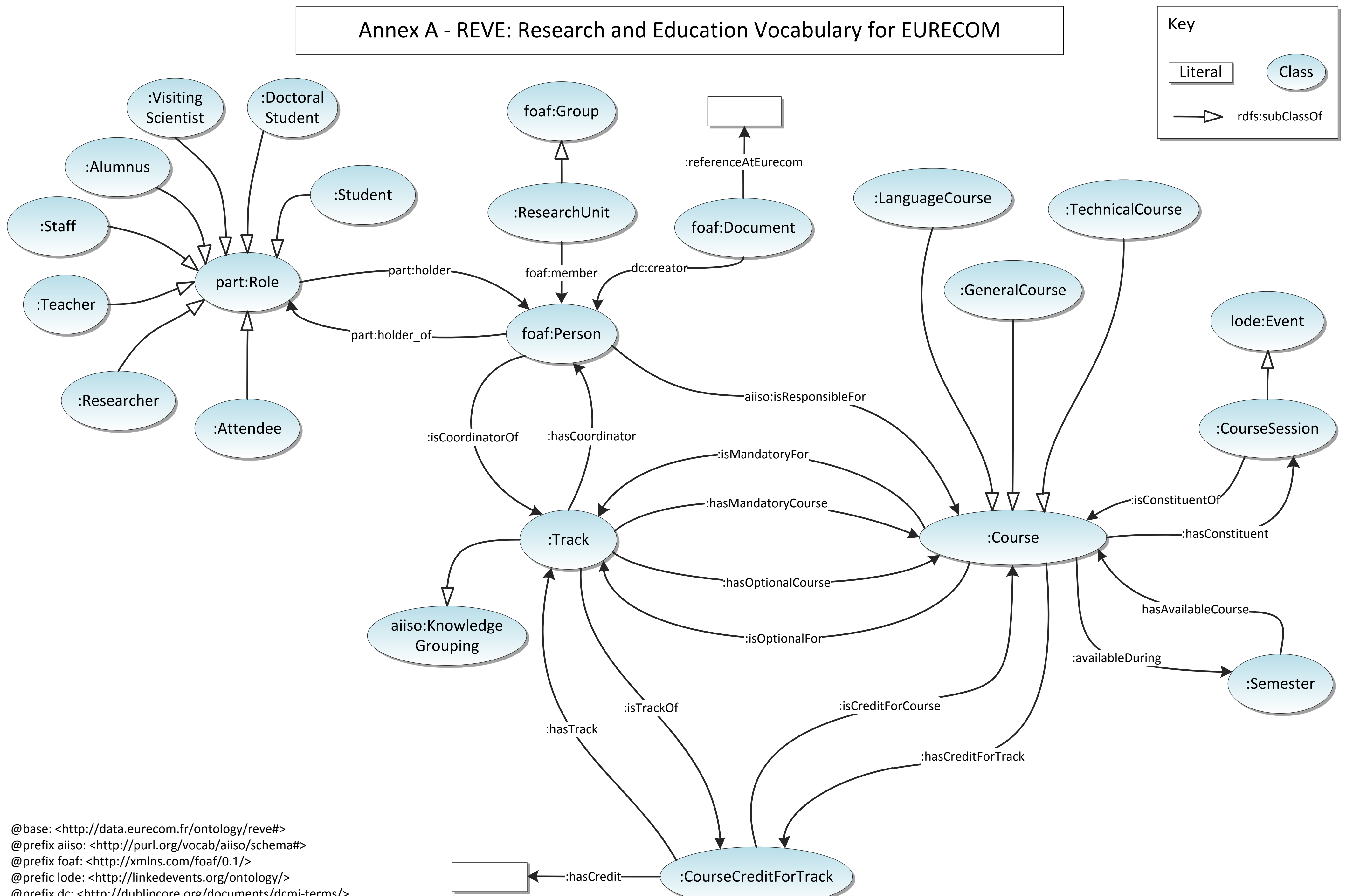
As the dataset gets enriched with data from internal and external sources, surely people will come up with some new ways to use the data, and build impressive applications. And why not, once the project is more mature and a public SPARQL endpoint is set up, organize a little contest among students rewarding the best applications ?

[3]http://data.open.ac.uk/applications/kmi-connections.php?cpaper=on

Bibliography

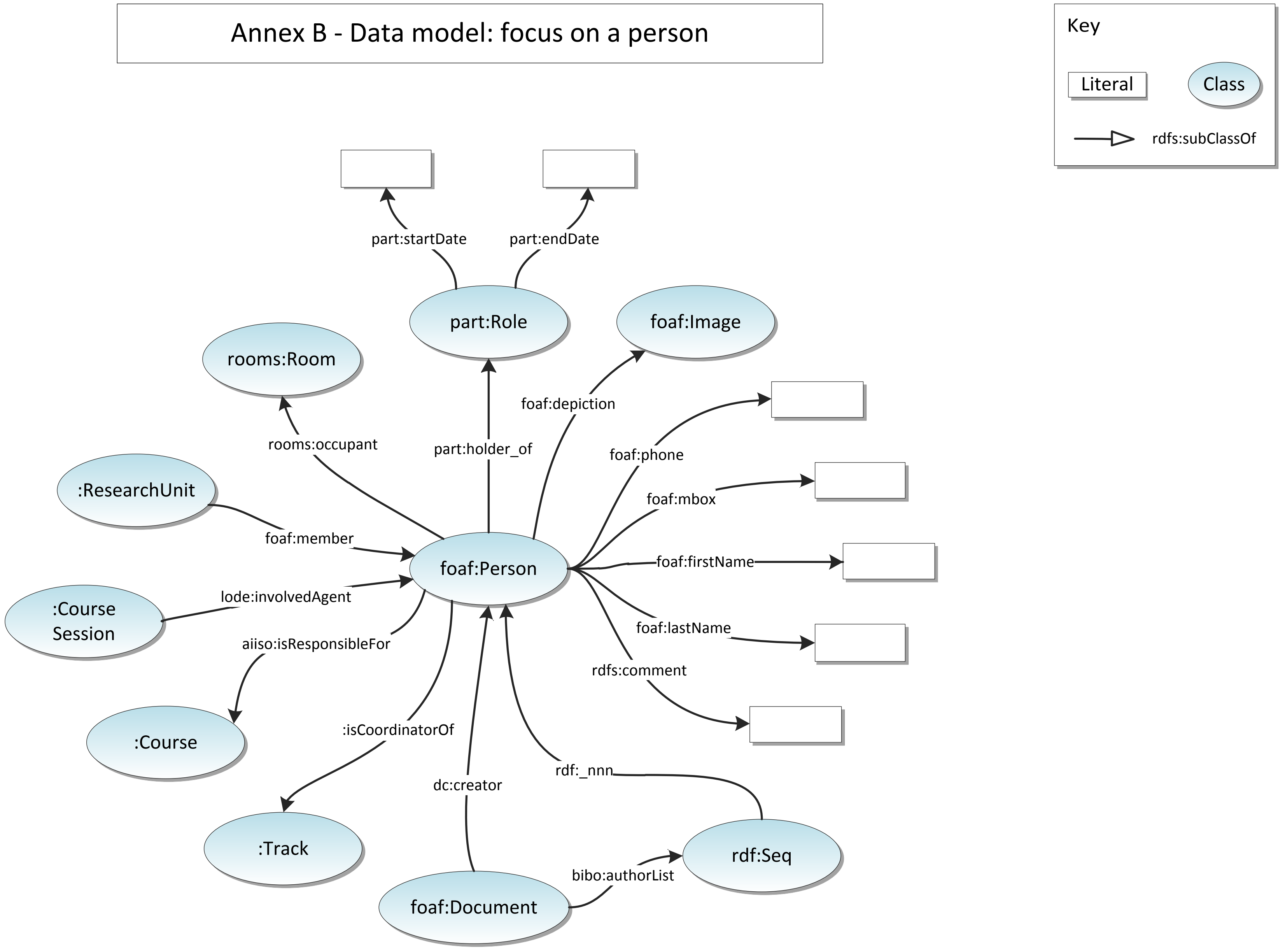
[1] Tom Heath and Christian Bizer. *Linked Data: Evolving the Web into a Global Data Space.* Morgan & Claypool, 1st edition, 2011.

Annex A - REVE: Research and Education Vocabulary for EURECOM



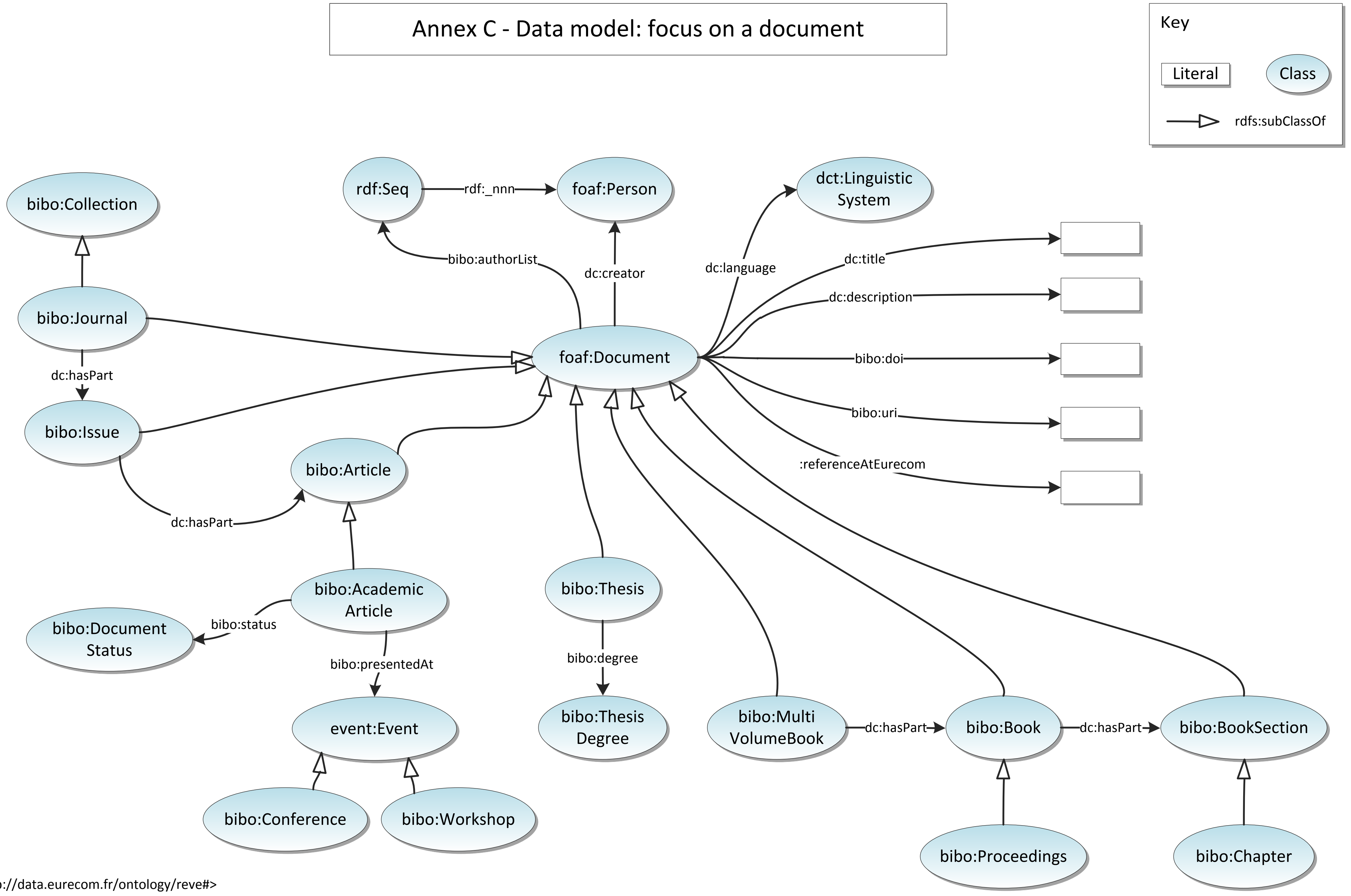
@base: <http://data.eurecom.fr/ontology/reve#>
@prefix aiiso: <http://purl.org/vocab/aiiso/schema#>
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefic lode: <http://linkedevents.org/ontology/>
@prefix dc: <http://dublincore.org/documents/dcmi-terms/>
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>

Annex B - Data model: focus on a person



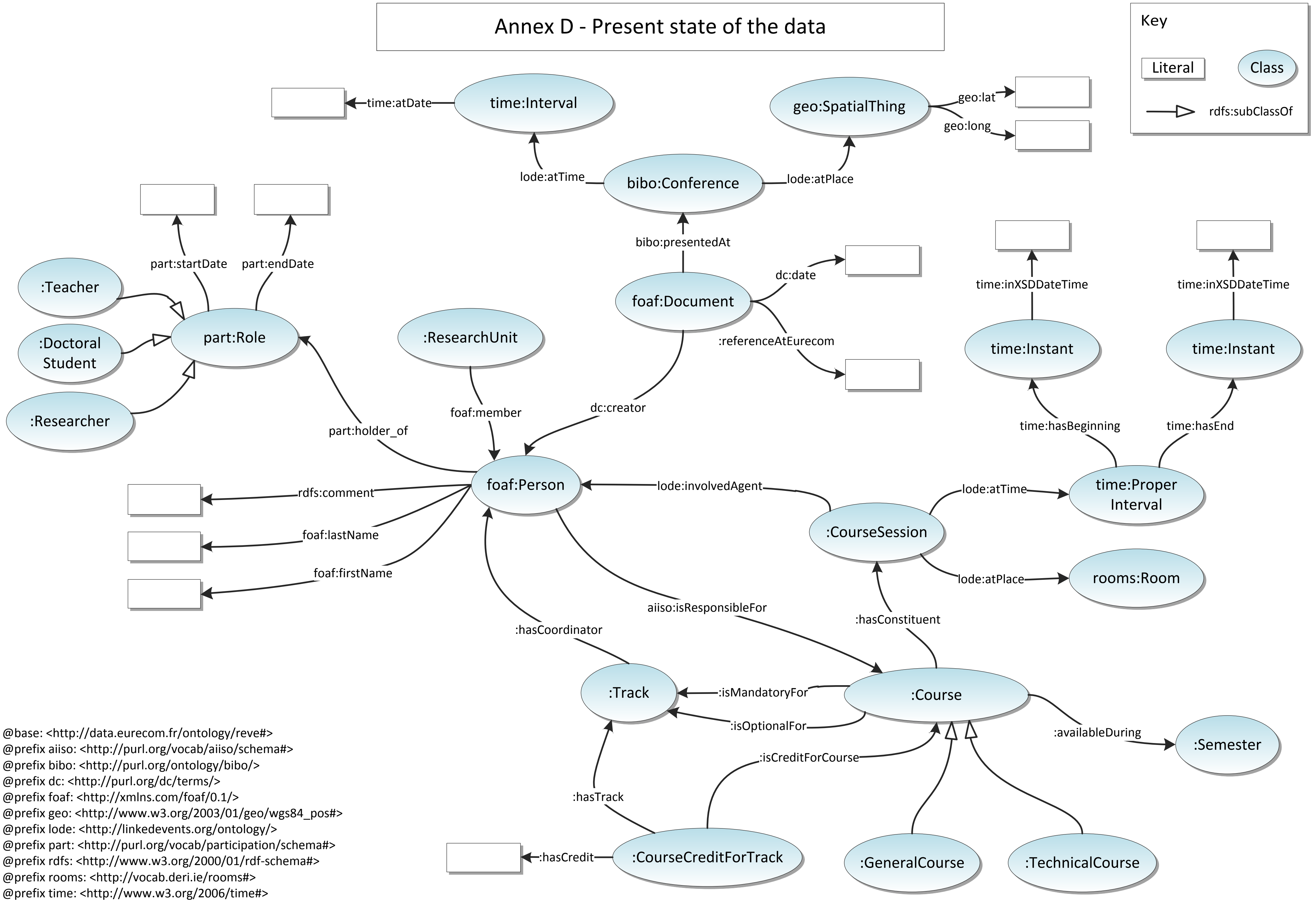
@base: <http://data.eurecom.fr/ontology/reve#>
@prefix aiiso: <http://purl.org/vocab/aiiso/schema#>
@prefix bibo: <http://purl.org/ontology/bibo/>
@prefix dc: <http://purl.org/dc/terms/>
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefix lode: <http://linkedevents.org/ontology/>
@prefix part: <http://purl.org/vocab/participation/schema#>
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
@prefix rooms: <http://vocab.deri.ie/rooms#>

Annex C - Data model: focus on a document



```
@base: <http://data.eurecom.fr/ontology/reve#>
@prefix bibo: <http://purl.org/ontology/bibo/>
@prefix dc: <http://purl.org/dc/terms/>
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefix lode: <http://linkedevents.org/ontology/>
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
```

Annex D - Present state of the data



@base: <http://data.eurecom.fr/ontology/reve#>
@prefix aiiso: <http://purl.org/vocab/aiiso/schema#>
@prefix bibo: <http://purl.org/ontology/bibo/>
@prefix dc: <http://purl.org/dc/terms/>
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
@prefix lode: <http://linkedevents.org/ontology/>
@prefix part: <http://purl.org/vocab/participation/schema#>
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
@prefix rooms: <http://vocab.deri.ie/rooms#>
@prefix time: <http://www.w3.org/2006/time#>